Document: WG14/N1247
Date: 2007/08/08
Project: TR 24732
Authors: Jim Thomas, Rich Peterson
Reply to: Rich Peterson <Rich.Peterson@hp.com>

Subject: completing specification for a, A output conversion

WG14 N1241 (TR 24732 draft of 2007/07/05) specifies the behavior of a, A output conversion with a decimal floating-point modifier when the precision modifier is missing. However, it does not specify the behavior when an explicit precision is given.

This proposal completes the specification of a, A output conversion to cover explicit as well as implicit precision. It is based on the idea of decimal a, A output showing the actual representation of a decimal floating-point number, which was a rationale given at the London meeting for choosing the letter “a” as the format specifier. If the precision modifier is sufficient, the conversion shows the actual representation of the decimal floating-point input. If not, the conversion shows the actual representation of the result of rounding the input to the given precision. Here an “actual” representation of a decimal number refers to one from which the components -- sign, coefficient, quantum exponent (5.2.4.2.2a) -- of the number can be determined.

Suggested TR changes: In 9.5, after the two bullets and before the Example, insert

----

If the precision modifier is present and at least as large as the precision p (5.2.4.2.2) of the decimal floating type (7 for _Decimal32, 16 for _Decimal64, and 34 for _Decimal128), the conversion is as if the precision modifier were missing. If the precision modifier is present and less than the precision p of the decimal floating type, the conversion first rounds the input, in the type, according to the current rounding direction, to the number of digits specified by the precision modifier, then converts the result as if the precision modifier were missing.

In 9.5, after Examples, append

To illustrate the effects of a precision modifier, the sequence

```
_Decimal32 x = 6543.00DF;  // represented by the triple ( 1, 654300, -2)
printf(“%Ha\n”, x);
printf(“%.6Ha\n”, x);
printf(“%.5Ha\n”, x);
printf(“%.4Ha\n”, x);
printf(“%.3Ha\n”, x);
printf(“%.2Ha\n”, x);
printf(“%.1Ha\n”, x);
```

results in

```
6543.00
6543.00
6543.0
6543
6.54e+3
6.5e+3
7e+3
```